1.

Code

#include <iostream>

using namespace std ;

class Point

{

public:

Point (int x = 0, int y = 0) : x(x), y(y)

{

cout << "Point constructed " ;

Display() ;

}

Point (Point &p) : x(p.x), y(p.y)

{

cout << "Point copy constructed " ;

Display() ;

}

~Point ()

{

cout << "Point destructed " ;

Display() ;

}

void Set(int a, int b) ;

void Set(Point &p) ;

void Display() ;

private:

int x, y ;

} ;

void Point::Set(int a, int b)

{

x = a ;

y = b ;

}

void Point::Set (Point &p)

{

x = p.x ;

y = p.y ;

}

void Point::Display()

{

cout << '(' << x << ',' << y << ')' << endl ;

}

class Line: public Point

{

public:

Line(int x = 0, int y = 0, double s = 0): Point(x, y), s(s)

{

cout << "Line constructed 1 " ;

Display() ;

}

Line(Point &p, double s): Point(p), s(s)

{

cout << "Line constructed 2 " ;

Display() ;

}

Line(Line &l): Point(l), s(l.s)

{

cout << "Line copy constructed " ;

Display() ;

}

~Line()

{

cout << "Line destructed " ;

Display() ;

}

void Set(int x, int y, double s) ;

void Set(Point &p, double s) ;

void Display() ;

private:

double s ;

} ;

void Line::Set(int x, int y, double s)

{

Point::Set(x, y) ;

this->s = s ;

}

void Line::Set(Point &p, double s)

{

Point::Set(p) ;

this->s = s ;

}

void Line::Display()

{

cout << "Slope=" << s << ' ' ;

Point::Display() ;

}

int main()

{

Point a ;

a.Display() ;

a.Set(2, 3) ;

a.Display() ;

Line l1(0, 0, 1), l2(a, 4), l3(l2) ;

l1.Display() ;

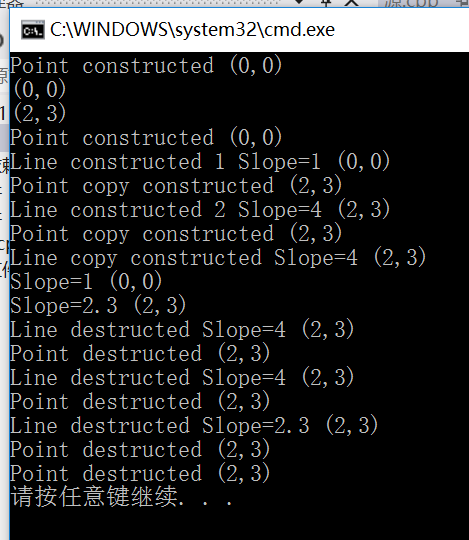
l1.Set(a, 2.3) ;

l1.Display() ;

return 0 ;

}

Result:



2.

Code:

#include <iostream>

using namespace std ;

class Point

{

public:

Point (int x = 0, int y = 0) : x(x), y(y)

{

cout << "Point constructed " ;

Display() ;

}

Point (Point &p) : x(p.x), y(p.y)

{

cout << "Point copy constructed " ;

Display() ;

}

~Point ()

{

cout << "Point destructed " ;

Display() ;

}

void Set(int a, int b) ;

void Set(Point &p) ;

void Display() ;

protected:

int x, y ;

} ;

void Point::Set(int a, int b)

{

x = a ;

y = b ;

}

void Point::Set (Point &p)

{

x = p.x ;

y = p.y ;

}

void Point::Display()

{

cout << '(' << x << ',' << y << ')' << endl ;

}

class Line: public Point

{

public:

Line(int x = 0, int y = 0, double s = 0): s(s)

{

this->x = x ;

this->y = y ;

cout << "Line constructed 1 " ;

Display() ;

}

Line(Point &p, double s): Point(p), s(s)

{

cout << "Line constructed 2 " ;

Display() ;

}

Line(Line &l): Point(l), s(l.s)

{

cout << "Line copy constructed " ;

Display() ;

}

~Line()

{

cout << "Line destructed " ;

Display() ;

}

void Set(int x, int y, double s) ;

void Set(Point &p, double s) ;

void Display() ;

private:

double s ;

} ;

void Line::Set(int x, int y, double s)

{

this->x = x ;

this->y = y ;

this->s = s ;

}

void Line::Set(Point &p, double s)

{

Point::Set(p) ;

this->s = s ;

}

void Line::Display()

{

cout << "Slope=" << s << ' ' ;

Point::Display() ;

}

int main()

{

Point a ;

a.Display() ;

a.Set(2, 3) ;

a.Display() ;

Line l1(0, 0, 1), l2(a, 4), l3(l2) ;

l1.Display() ;

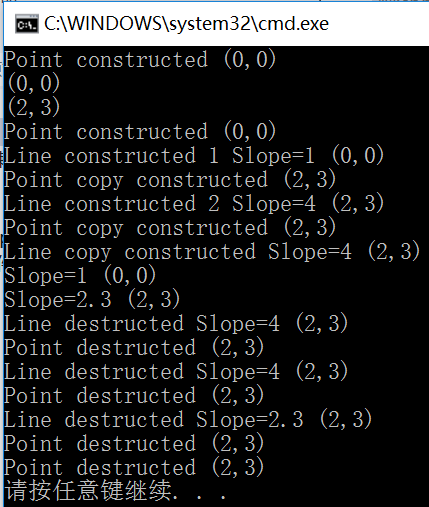
l1.Set(a, 2.3) ;

l1.Display() ;

return 0 ;

}

Result:



实在不知道怎么简化…

3.

#include <iostream>

#include <cmath>

using namespace std ;

class Point

{

public:

Point (double x = 0, double y = 0) : x(x), y(y)

{

cout << "Point constructed " ;

Print() ;

}

Point (Point &p) : x(p.x), y(p.y)

{

cout << "Point copy constructed " ;

Print() ;

}

~Point ()

{

cout << "Point destructed " ;

Print() ;

}

void Set(double a, double b) ;

void Set(Point &p) ;

void Print() ;

double distance(Point &p2) ;

friend class Triangle ;

private:

double x, y ;

} ;

double Point::distance(Point &p2)

{

return sqrt((x - p2.x) \* (x - p2.x) + (y - p2.y) \* (y - p2.y)) ;

}

void Point::Set(double a, double b)

{

x = a ;

y = b ;

}

void Point::Set (Point &p)

{

x = p.x ;

y = p.y ;

}

void Point::Print()

{

cout << '(' << x << ',' << y << ')' << endl ;

}

class Triangle: public Point

{

public:

Triangle(Point p1, Point p2, Point p3): Point(p1), p1(p2), p2(p3)

{

cout << "Triangle constructed: " << endl ;

Print() ;

}

Triangle(Triangle &t): Point(t), p1(t.p1), p2(t.p2)

{

cout << "Triangle copy constructed: " << endl ;

Print() ;

}

void Set(Point &p1, Point &p2, Point &p3) ;

double Area() ;

Point Gpoint() ;

void Print() ;

private:

Point p1, p2 ;

} ;

double Triangle::Area()

{

double a = Point::distance(p1), b = Point::distance(p2), c = p1.distance(p2), p = a + b + c ;

return sqrt(p \* (p - a) \* (p - b) \* ( p - c)) ;

}

Point Triangle::Gpoint()

{

double a = (Point::x + p1.x + p2.x) / 3, b = (Point::y + p1.y + p2.y) / 3 ;

return Point(a, b) ;

}

void Triangle::Set(Point &p1, Point &p2, Point &p3)

{

Point::Set(p1) ;

(this->p1).Set(p2) ;

(this->p2).Set(p3) ;

}

void Triangle::Print()

{

cout << "The points are:" << endl ;

cout << "P1: " ;

Point::Print() ;

cout << "P2: " ;

p1.Print() ;

cout << "P3: " ;

p2.Print() ;

}

int main()

{

Point p1(1, 2), p2(1, 3), p3(4, 2) ;

Triangle t1(p1, p2, p3) ;

t1.Print() ;

Triangle t2(t1) ;

t2.Print() ;

Point G1 = t1.Gpoint() ;

G1.Print() ;

t2.Set(p1, p2, G1) ;

t2.Print() ;

cout << "The area of t2 is " << t2.Area() << endl ;

return 0 ;

}

Result:

Point constructed (1,2)

Point constructed (1,3)

Point constructed (4,2)

Point copy constructed (4,2)

Point copy constructed (1,3)

Point copy constructed (1,2)

Point copy constructed (1,2)

Point copy constructed (1,3)

Point copy constructed (4,2)

Triangle constructed:

The points are:

P1: (1,2)

P2: (1,3)

P3: (4,2)

Point destructed (1,2)

Point destructed (1,3)

Point destructed (4,2)

The points are:

P1: (1,2)

P2: (1,3)

P3: (4,2)

Point copy constructed (1,2)

Point copy constructed (1,3)

Point copy constructed (4,2)

Triangle copy constructed:

The points are:

P1: (1,2)

P2: (1,3)

P3: (4,2)

The points are:

P1: (1,2)

P2: (1,3)

P3: (4,2)

Point constructed (2,2.33333)

(2,2.33333)

The points are:

P1: (1,2)

P2: (1,3)

P3: (2,2.33333)

The area of t2 is 5.76377

Point destructed (2,2.33333)

Point destructed (2,2.33333)

Point destructed (1,3)

Point destructed (1,2)

Point destructed (4,2)

Point destructed (1,3)

Point destructed (1,2)

Point destructed (4,2)

Point destructed (1,3)

Point destructed (1,2)

请按任意键继续. . .